REMARKS

Claims 1-19 and 24-27 were pending and presented for examination. In an Office Action dated November 20, 2007, all pending claims were rejected. In view of the Remarks that follow, Applicant respectfully requests that Examiner reconsider all outstanding objections and rejections, and withdraw them.

Response to Rejections Under 35 USC 103(a)

In the Office Action, Examiner rejected claims 1-19 and 24-27 under 35 USC § 103(a). Claims 1-13 and 24-26 were rejected over U.S. Patent No. 5,881,151 to Yamamoto in view of U.S. Patent No. 5,826,013 to Nachenberg. Claims 14-18 and 20-23 were rejected over Yamamoto and Nachenberg in view of U.S. Patent No. 5,734,908 to Chan. Claim 19 was rejected over Yamamoto, Nachenberg, and Chan in view of U.S. Patent Publication No. 2004/0221279 to Lovett. Claim 27 was rejected over Yamamoto in view of Lovett. These rejections are respectfully traversed.

Claim 1 recites a computer-implemented method for determining whether computer code contains malicious code, the method comprising:

> identifying **computer code suspected of currently containing malicious code**;
> **optimizing the identified computer code** to produce optimized code; and
> subjecting the optimized code to a malicious code detection protocol; and
> responsive to the malicious code detection protocol detecting malicious code in the optimized code, declaring a confirmation that the computer code contains malicious code.
> (emphasis added)

Claim 6 recites a computer-implemented method for determining whether computer code contains malicious code, the method comprising:

> identifying **computer code suspected of currently containing malicious code, the computer code having a decryption loop and a body**;
> **optimizing the decryption loop** to produce optimized loop code;
> performing a malicious code detection procedure on the optimized loop code;
> **optimizing the body** to produce optimized body code;
> subjecting the optimized body code to a malicious code detection protocol; and
> responsive to the malicious code detection procedure detecting malicious code in the optimized loop code or the malicious code detection protocol detecting malicious code in the optimized body code, declaring a confirmation that the computer code contains malicious code.
> (emphasis added)

As can be seen, claims 1 and 6 recite identifying computer code suspected of currently containing malicious code and optimizing that computer code to produce optimized code. In claim 6, the computer code has a decryption loop and body, both of which are optimized. The optimized code is then subjected to a malicious code detection protocol, and a confirmation of malicious code is declared responsive to detecting malicious code. The claimed invention beneficially optimizes the code to simplify it so that malicious code detection protocols can be more efficiently and effectively applied to the code. For example, a virus may contain intentionally complexified code that makes virus detection protocols slow or inaccurate. The claimed invention first optimizes the code so that the detection protocols can be more successful. Claim 24 contains similar language to claim 1. All arguments regarding claim 1 presented below apply equally to claim 24.

Claim 1 and 6 are not disclosed by Yamamoto and Nachenberg. Yamamoto discloses compiling and optimizing a virus-free source program and adding virus diagnosing (object) code to the resulting program (object) code (Yamamoto, col. 4, lines 21-46). The virus diagnosing code can be run later to verify that the program code has not been modified by a virus subsequent to being compiled (Yamamoto, col. 6, lines 38-50). The diagnosing code operates under the assumption that the program code is free of viruses since it was compiled and optimized from the known source program. Yamamoto is not concerned with optimizing code that is currently suspected of currently containing malicious code. Nachenberg describes a method for detecting a polymorphic virus using emulation, similar to the method described in the Background Art of the present specification. Nachenberg does not disclose optimizing code that is suspected of currently containing malicious code.

Accordingly, the references do not show "optimizing the identified computer code ..." in claim 1. Examiner cites Yamamoto, col. 4, lines 51-55, col. 5, lines 26-38, and the code optimizing portion 38 for disclosing this element. These portions merely describe standard code optimizations, as known in the art, being performed on the intermediate file 36. However, as described above, these optimizations are not performed on code that is suspected of currently containing malicious code. The intermediate file 36 is created from the known source program 10. The "identified computer code" recited in this element of claim 1 explicitly refers to "computer code suspected of currently containing malicious code." Yamamoto teaches away from "optimizing the identified computer code ..." by disclosing optimizing known clean code rather than optimizing code suspected of currently containing malicious code.

In rejecting claim 6, Examiner does not address "identifying computer code suspected of currently containing malicious code, the computer code having a decryption loop and a body," as recited in the claim. Examiner cites col. 6, line 54 to col. 7, line 8 and the polymorphic anti-virus module (PAM) 200 of Nachenberg as disclosing "optimizing the decryption loop to produce optimized loop code" and "optimizing the body to produce optimized body code" However, this portion of Nachenberg merely discloses emulating, not optimizing, a portion of code to determine if the code contains a virus. Such emulation may reveal a virus decryption loop. While Nachenberg does mention "[reducing] the number of file instructions that must be emulated in order to determine whether a target file is infected by a virus" at col. 6, lines 57-59, this reducing is not performed through code optimization but rather through various virus exclusion methods as described in col. 6, line 59 to col. 7, line 8.

Based on the above remarks, Applicant submits that for at least these reasons claims 1, 6, and 24 and dependent claims 2-5, 7-19, and 25-26 are patentably distinguishable over the cited references. Therefore, Applicant respectfully requests that Examiner reconsider the rejection, and withdraw it. Chan and Lovett do not remedy the deficiencies of Yamamoto and Nachenberg with respect to these arguments. Chan is concerned with source code optimization methods that better utilize resources on the underlying machine. Lovett discloses dead code elimination when translating code from one processor instruction set to another. None of these references are concerned with optimizing code suspected of currently containing malicious code.

Claim 5, dependent on claim 1, recites "wherein the computer code is polymorphic code comprising a decryption loop and a body; and the optimizing step comprises optimizing

just the decryption loop." Examiner cites col. 6, line 54 to col. 7, line 8 and the polymorphic anti-virus module (PAM) 200 of Nachenberg as disclosing "the optimizing step comprises optimizing just the decryption loop." However, as discussed above in the context of claim 6, this portion of Nachenberg does not disclose the claimed element. Applicant submits that claim 5 is patentably distinguishable over the cited references for this reason in addition to the reasons given above.

Claim 27 recites a method for determining whether computer code contains malicious code, the method comprising:

> performing a dead code elimination procedure on the computer code;
> noting the amount of dead code eliminated during the dead code elimination procedure; and
> when the **amount of dead code eliminated during the dead code elimination procedure exceeds a preselected dead code threshold**, declaring a suspicion of malicious code in the computer code.
> (emphasis added)

As can be seen, the claim recites performing a dead code elimination procedure on the computer code, noting the amount of dead code eliminated during the procedure, and declaring a suspicion of malicious code in the computer code when this amount exceeds a threshold. The claimed invention enables the detection of suspicious code by determining that the code contains a certain amount of dead code, which is often found in malicious code.

Claim 27 is not disclosed by the combination of Yamamoto and Lovett. As discussed above, Yamamoto discloses optimizing only clean source code, not optimizing computer code to determine if there was malicious code in the computer code prior to the optimization (dead code elimination is a type of optimization). Lovett discloses dead code elimination but is not concerned with declaring a suspicion of malicious code based on the amount of dead code eliminated.

Accordingly, the references do not disclose "when the amount of dead code eliminated during the dead code elimination procedure exceeds a preselected dead code threshold, declaring a suspicion of malicious code in the computer code." Paragraphs [0133], [0144], [0091], and [0098] cited by Examiner merely mention thresholds used in other contexts, such as an execution count threshold for group block generation or a profiling metric threshold for group block construction. These are not thresholds of amounts of dead code, and the thresholds are not used to declare a suspicion of malicious code. In paragraph [0091] of Lovett, dead code elimination is **triggered by** a profiling metric exceeding a certain threshold, teaching away from the claimed invention.

In the "Response to Arguments" section of the Office Action, Examiner further points to Lovett [0107] and Yamamoto, col. 6, lines 31-37, as disclosing the above element. However, Lovett [0107] merely discloses recording global dead code transformations in subject registers in order to prune IR (Intermediate Representation) trees. Yamamoto, col. 6, lines 31-37, merely discloses interrupting execution and creating an output after a virus is detected. Examiner also states that "dead code elimination is a profiling metric." However, Examiner does not provide support for this statement. Lovett provides examples of profiling metrics in paragraph [0093] and none of the examples include dead code elimination.

Based on the above remarks, Applicant respectfully submits that for at least these reasons claim 27 is patentably distinguishable over the cited references. Therefore, Applicant respectfully requests that Examiner reconsider the rejection, and withdraw it.

Applicant invites Examiner to contact Applicant's representative at the number provided below if Examiner believes it will help expedite furtherance of this application.

Respectfully Submitted,
Frederic Perriot

Date: January 22, 2008

By: /Nikhil Iyengar/

Nikhil Iyengar, Reg. No. 60,910
Attorney for Applicant
Fenwick & West LLP
801 California Street
Mountain View, CA 94041
Tel.: (650) 335-7627
Fax: (650) 938-5200